دانشکده مهندسی کامپیوتر

استاد درسی: دکتر اعرابی و دکتر ابوالقاسمی

پاییز ۱۴۰۰

گزارش پیاده سازی مدل
درس یادگیری ماشین

کسرا سینایی، امیرحسین افخمی
شماره دانشجویی: ۸۱۰۶۹۶۲۵۴، ۸۱۰۶۹۶۲۰۶

گزارش نهایی

---

۱ فیچر اکسترکشن

برای رسیدن به یک مدل مناسب برای طبقه بندی و همچنین خوشه بندی، مهم ترین عامل، داشتن فیچر مناسب است. برای استخراج فیچرهای مناسب از آهنگ‌های موجود سعی شد تا تمامی فیچرهای اصلی ممکن استخراج شود. در مجموع ۹۷ فیچر از آهنگ‌های موجود در دیتاست استخراج شد که در ادامه آورده شده‌اند. لازم به ذکر است که به طور کلی دو رویکرد برای استخراج ویژگی از داده‌های صوتی وجود دارد که یکی از آن‌ها مطابق رویکردی است که در این پروژه اتخاذ شده است ودیگری تبدیل داده‌های صوتی به صورت فریم‌های تصویری و استفاده از این تصاویر برای تسک‌های یادگیری ماشین است. از آن جایی که استفاده از شبکه‌های کانولوشنی در این پروژه مجاز نیست، در این جا از این روش استفاده نشد.
در فیچرهای زیر، مقادیر واریانس و میانگین به عنوان فیچر برای هر یک از آهنگ‌ها در نظر گرفته شد.

- Chroma STFT
- Spectral Centroid
- Spectral Rolloff
- Spectral Bandwidth
- Zero Crossing Rate
- RMS Value for Each Frame
- Roll-Off Frequency
- Tempo
- Pulse
- Harmony
- Mel-Frequency Cepstral Coefficients: در این فیچر میانگین و واریانس ۲۰ MFCC اول به عنوان فیچر در نظر گرفته شدند.
- Mel-Scaled Spectrogram: در این جا نیز همچون MFCC، میانگین و واریانس ۲۰ فیچر اول به عنوان فیچر در نظر گرفته شدند.

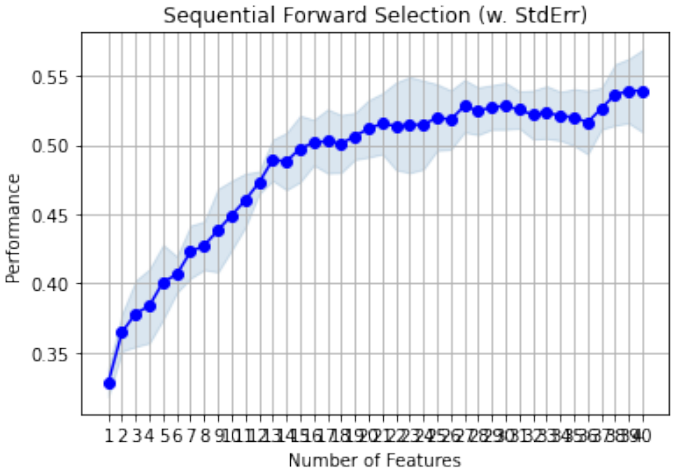
در نهایت برای هر یک از آهنگ‌ها به طور جداگانه فیچرهای فوق استخراج شدند و در یک فایل CSV ذخیره شدند. در پایان باید به این نکته توجه داشت که تمامی فیچرهای اکسترکت شده مناسب نیستند و به اصطلاح فیچر خوبی برای تسک کلسیفیکیشن به حساب نمیایند، چرا که Discriminability لازم را ایجاد نمی‌کنند. برای این موضوع در قسمت‌های بعدی، تمهیداتی همچون فیچر سلکشن و فیچر ریداکشن اندیشیده شده است.

۲ پیش پردازش

به دلیل طولانی شدن استخراج ویژگی‌ها از فایل خام داده‌ها، هر کدام از ژانرها با تابع read folder از پوشه‌های جداگانه باز شدند و فیچرهای به دست آمده از آن‌ها در یک فایل csv جدا ذخیره شدند. (یک بار که از تابع read songs استفاده شد گوگل کولب دستور رانیمه کاره اجرا کرد.) بنابراین اولین مرحله از پیش پردازش داده ادغام سطرهای این فایل‌ها در یک دیتافریم است. برای راحتی کار با داده‌ها و خواندن فایل‌های اکسل از کتابخانه pandas کمک گرفته شده است. قبل از تقسیم مجموعه آموزش و تست، جداگانه هر یک از گروه‌ها را تقسیم می‌کنیم و سپس همه آن‌ها را در آرایه‌های numpy می‌ریزیم. به این ترتیب هم در مجموعه‌ی تست و هم در مجموعه‌ی آموزش داده‌ها بالانس خواهند بود.
یکی از مسائل مهم در کار با داده‌ها نرمال بودن یا استاندارد بودن ابعاد مختلف هر یک از سمپل‌ها می‌باشد. در این پروژه ادهای زیادی از سیگنال به دست آمده از هر آهنگ استخراج شده است و هر کدام بیانگر ویژگی خاصی هستند. اسکیل و بازه تغییر آن‌ها نیز متفاوت می‌باشد. برای بهبود عملکر الگوریتم‌های خوشه بندی و یا طبقه بندی قبل از انجام هر کدام از این الگوریتم‌ها تمامی سمپل‌های مجموعه آموزش و تست استاندارد شده‌اند. برای این کار نیز از انتقال standard scalar کتابخانه sci-kit learn استفاده شده است.
برای احتیاط هر دو آرایه استاندارد شده و استاندارد نشده را نگه می‌داریم اما برای آموزش و تست از از داده‌های استاندارد استفاده خواهیم کرد.

۱.۲ Feature Selection

برای پیدا کردن بهترین فیچرهای موجود برای رسیدن به بیشترین میزان دقت طبقه بند، در این جا از تکنیک Sequential Forward Selection استفاده شد. برای بهینه تر کردن انتخاب فیچرها، از Floating نیز استفاده شد. معیار انتخاب ویژگی‌ها در فرایند فیچر سلکشن، میزان دقت طبقه بند SVM با کرنل Linear در نظر گرفته شد. به این صورت ۴۰ فیچر اول انتخاب شدند، همچنین برای انتخاب این فیچرها از ۵ Cross Validation نیز استفاده شد. در ادامه نمودار مربوط به دقت مدل به ازای فیچر انتخابی آورده شده است.



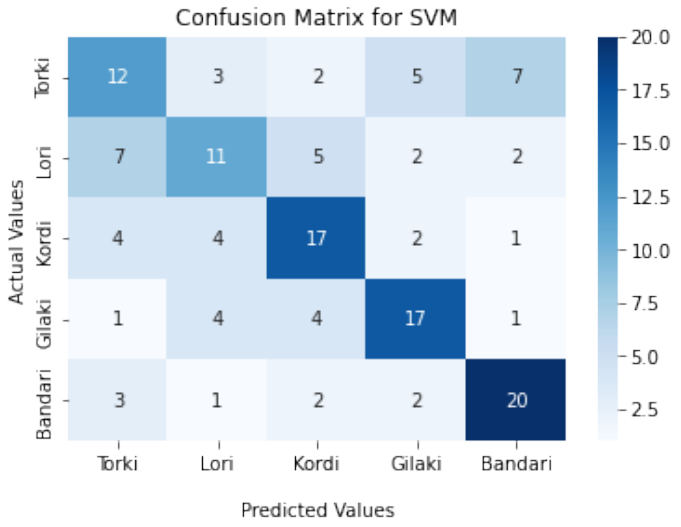
شکل ۱: انتخاب فیچرها با استفاده از SFS

۳ طبقه بندی

برای طبقه بندی از ۳ الگوریتم مختلف استفاده شد. الگوریتم‌های مورد استفاده شامل SVM، KNN و MLP بودند. نتایج حاصل برای هر یک از ۳ الگوریتم طبقه بند در ادامه آورده شده است.

۱.۳ SVM:

الگوریتم soft-SVM با استفاده از کرنل‌های مختلف پیاده سازی شد. بهترین دقت حاصل با در نظر گفتن کرنل Linear و مقدار ترم Regularization ۵ بود که برابر با ۵۵ درصد شد. Confusion Matrix مربوط به این طبقه بند در ادامه آورده شده است.



شکل ۲: ماتریس کانفوژن برای الگوریتم SVM

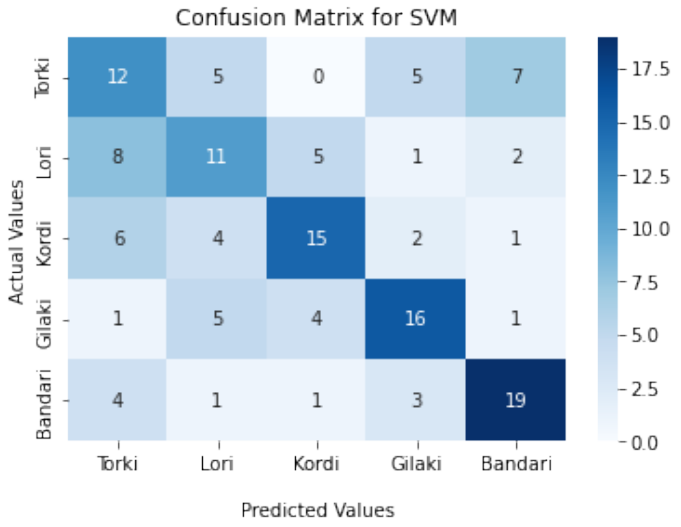
میزان دقت، Precision، Recall و F1-score برای این طبقه بند نیز در ادامه آورده شده است.
نتایج حاصل به دست آمده با استفاده از کرنل RBF برای الگوریتم SVM نیز در ادامه آورده شده است.

accuracySVM: 0.5539568345323741

classificationSVMreport:

	precision	recall	f1-score	support
0.0	0.44	0.41	0.43	29
1.0	0.48	0.41	0.44	27
2.0	0.57	0.61	0.59	28
3.0	0.61	0.63	0.62	27
4.0	0.65	0.71	0.68	28
accuracy			0.55	139
macro avg	0.55	0.55	0.55	139
weighted avg	0.55	0.55	0.55	139

شکل ۳: میزان دقت، Precision، Recall و F1-score برای الگوریتم SVM



شکل ۴: ماتریس کانفوژن به دست آمده با استفاده از کرنل RBF برای الگوریتم SVM

accuracySVM: 0.5251798561151079

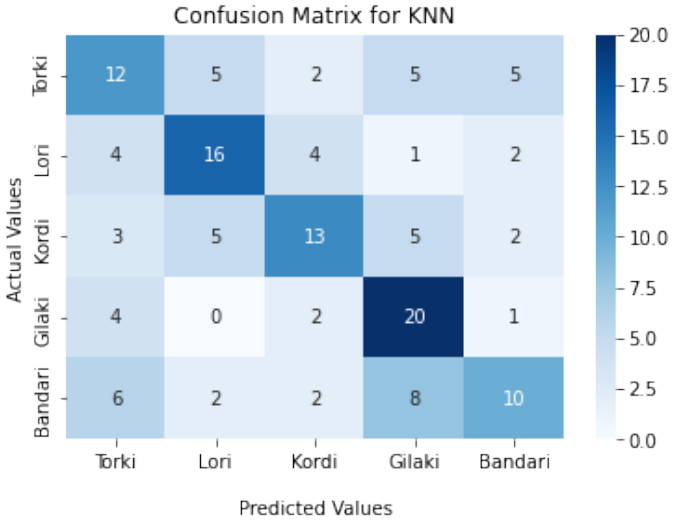
classificationSVMreport:

	precision	recall	f1-score	support
0.0	0.39	0.41	0.40	29
1.0	0.42	0.41	0.42	27
2.0	0.60	0.54	0.57	28
3.0	0.59	0.59	0.59	27
4.0	0.63	0.68	0.66	28
accuracy			0.53	139
macro avg	0.53	0.53	0.53	139
weighted avg	0.53	0.53	0.53	139

شکل ۵: نتایج حاصل به دست آمده با استفاده از کرنل RBF برای الگوریتم SVM

۲.۳ KNN:

در این الگوریتم، بیشترین میزان دقت با استفاده از در نظر گرفتن نزدیک ترین همسایه بیشترین میزان دقت بدست آمد که برابر با ۵۱ درصد شد. متریک در نظر گرفته شده برای الگوریتم نزدیک ترین همسایه، فاصله اقلیدسی در نظر گرفته شد. نتایج حاصل از طبقه بند نزدیک ترین همسایه در ادامه آورده شده است.



شکل ۶: ماتریس کانفوژن برای الگوریتم KNN

accuracyKNN: 0.5107913669064749

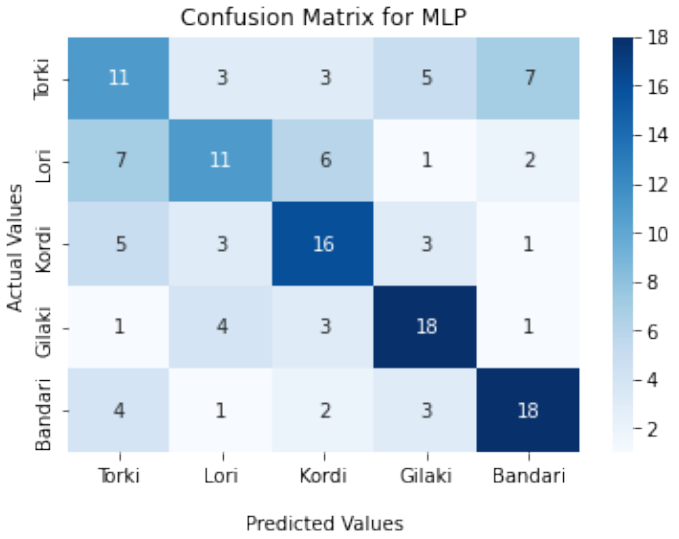
classificationKNNreport:

	precision	recall	f1-score	support
0.0	0.41	0.41	0.41	29
1.0	0.57	0.59	0.58	27
2.0	0.57	0.46	0.51	28
3.0	0.51	0.74	0.61	27
4.0	0.50	0.36	0.42	28
accuracy			0.51	139
macro avg	0.51	0.51	0.51	139
weighted avg	0.51	0.51	0.50	139

شکل ۷: نتایج حاصل به دست آمده برای الگوریتم KNN

۳.۳ :MLP

ساختار شبکه در نظر گرفته شده برای این الگوریتم، یک ساختار سه لایه با ۱۲۸، ۶۴ و ۳۲ نورون در لایه پنهان است. همچنین تابع فعالساز ReLu برای نورون‌های شبکه طراحی شده در نظر گرفته شد. بین تمام لایه‌های پنهان نیز لایه دراپ اوت جهت جلوگیری از اورفیت مدل قرار داده شد. مدل در ۱۰۰۰ ایپاک و در بچ‌های ۶۴ تایی آموزش داده شد. نتایج حاصل در ادامه آورده شده است.



شکل ۸: ماتریس کانفوژن برای الگوریتم MLP

accuracyMLP: 0.5323741007194245

classificationMLPreport:

	precision	recall	f1-score	support
0.0	0.39	0.38	0.39	29
1.0	0.50	0.41	0.45	27
2.0	0.53	0.57	0.55	28
3.0	0.60	0.67	0.63	27
4.0	0.62	0.64	0.63	28
accuracy			0.53	139
macro avg	0.53	0.53	0.53	139
weighted avg	0.53	0.53	0.53	139

شکل ۹: نتایج حاصل به دست آمده برای الگوریتم MLP

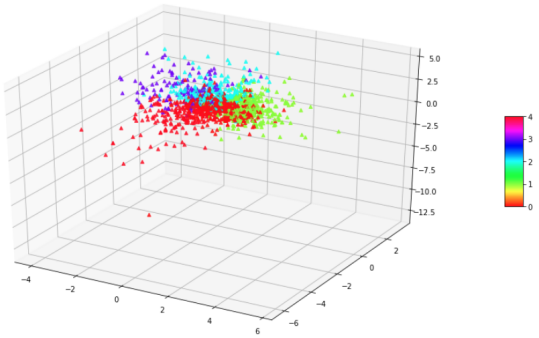
میتوان دید که بیشترین دقت به دست آمده نزدیک ۵۵ درصد بود که دقت بسیار پایینی است. علت آن را می‌توان در فیچرهای کاهش یافته و به نمایش در آمده در PCA و LDA دید. همان طور که مشخص است، فیچرها از Discriminability چندانی برخوردار نیستند و کم بودن دیتا نیز عاملی افزاینده بر این مشکل است.

۴ خوشه بندی

برای ایجاد یکپارچگی بین الگوریتم‌های خوشه بندی همه آن‌ها را مانند قسمت طبقه بندی به صورت متدهای یک کلاس پیاده سازی کردیم تا فرآیند آزمایش و ارزیابی آن‌ها ساده تر شود.
توابع پیاده سازی شده در این کلاس هر سه نوع الگوریتم معرفی شده در درس را اجرا می‌کنند. الگوریتم K-Means به همراه الگوریتم‌هایheirarchial و density base در این کلاس موجود می‌باشند.
برای ارزیابی الگوریتم‌ها با توجه به اینکه لیبل واقعی و درست آهنگ‌ها را می‌دانیم می‌توانیم از Purity Measure که در کلاس درس معرفی شده است استفاده کنیم. علاوه بر این معیار یکی دیگر از معیارهای پر کاربرد برای الگوریتم‌های کلاسترینگ، Silhouette Scores است. این معیار در بازه ۱ و ۱- قرار دارد. هر چه به ۱ نزدیک تر باشد به این معنا است که خوشهذهای به دست آمده از یکدیگر بیشتر فاصله داشته و بهتر جدا شده‌اند. در مقابل هر چه این معیار به ۱- نزدیک تر باشد به این معنی است که خوشه‌ها به یکدیگر نزدیک تر بوده و کیفیت خوشه بندی در این مسئله کمتر می‌باشد.
پس از اجرای هر الگوریتم می‌توان با استفاده از متد visualize هم معیارهای خوشه بندی ذکر شده را ملاحظه کرد و هم تصویر سه بعدی کاهش بعد یافته با استفاده از LDA خوشه‌های به دست آمده را ترسیم کرد. توابع مورد نیاز برای محاسبه معیارهای نام برده شده جهت ارزیابی الگوریتم کلاسترینگ در کتابخانه sci-kit به نام‌های silhouette score و homogeniety score موجود می‌باشند.
با توجه به نمایش داده‌ها در ابعاد پایین تر می‌توان نتیجه گرفت که الگوریتم‌های خوشه بندی در این مسئله عملکرد خوبی نداشته باشند، زیرا داده‌ها کاملا در یکدیگر فرو رفته اند و در ناحیه میانی از فضا چگالی بسیار بالایی از داده‌های مربوط به گروه‌های مختلف را شاهد هستیم.

۱.۴ K-Means

این الگوریتم بهترین نتیجه را در بین سایر الگوریتم‌های خوشه بندی در پی داشت. از کتابخانه scikit مجددا در این قسمت استفاده شده است. همچنین برای بهبود عملکرد این الگوریتم لازم است داده‌ها حتما در فضایی کاهش بعد داده شده در الگوریتم استفاده شوند. نتایج این الگوریتم در شکل ۱۰ نشان داده شده است.

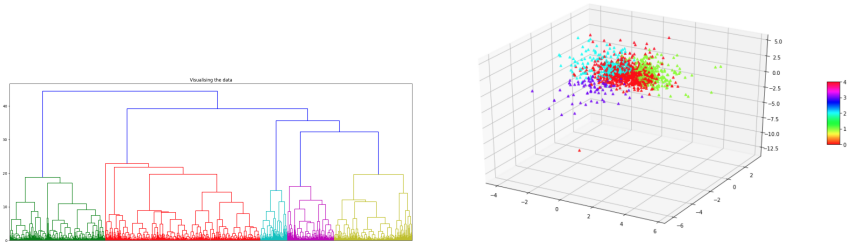


شکل ۱۰: نتیجه خوشه بندی با روش K Means

۲.۴ Heirarchial Methods

در این قسمت با استفاده از روش bottom-up و معیار ward برای تعیین فاصله خوشه‌ها الگوریتم خوشه بندی Heirarchial را اجرا می‌کنیم. برای فاصله نقاط، فاصله اقلیدسی در نظر گرفته شده است اما امکان استفاده از سایر فاصله‌های موجود در کتابخانه scipy و sklearn استفاده کرد. برای پیاده سازی الگوریتم اصلی می‌توان از تابع AgglomerativeClustering استفاده کرد و همچنین برای رسم dendogram از تابع shc کتابخانه scipy کمک گرفت.

خوشه‌های به دست آمده با این روش و همچنین دندوگرام الگوریتم در شکل ۱۱ نشان داده شده‌اند.



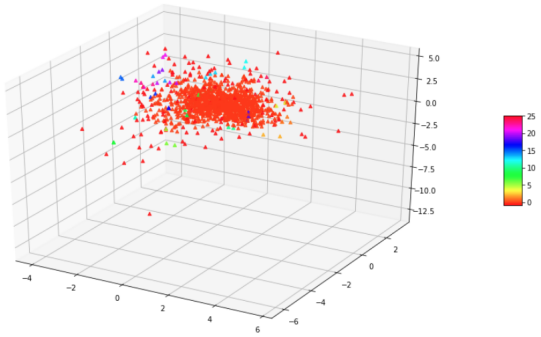
(آ) خوشه‌های به دست آمده (ب) Dendogram

شکل ۱۱: نتیجه خوشه بندی با معیار ward

۳.۴ DBSCAN

به دلیل چگالی بالا و تفکیک پذیری کم داده‌ها این الگوریتم نتایج جالبی به همراه نداشته و نتوانسته است داده‌ها را به حداقل تعداد کلاس مورد نظر (۵) تقسیم بندی کند و تنها تعداد محدودی داده به عنوان outlier شناسایی شدند.

نتایج بد این طبقه بندی نیز در شکل ۱۲ نشان داده شده‌اند.



شکل ۱۲: نتیجه خوشه بندی با روش DBSCAN

در جدول زیر بین تمامی روش‌های خوشه بندی پیاده سازی شده مقایسه‌ای صورت گرفته است که بر اساس معیارهای purity و silhouette score می‌باشند. همانطور که مشاهده می‌شود به هیچ وجه نتایج قابل قبولی به دست نامده است.

Measure	K-Means	Hierarchical (Complete)	Hierarchical (Average)	Hierarchical (Ward)	DBSCAN
Silhouette	0.204	0.110	0.520	0.166	-0.164
Purity	0.194	0.109	0.167	0.163	0.106

۵ Data Augmentation

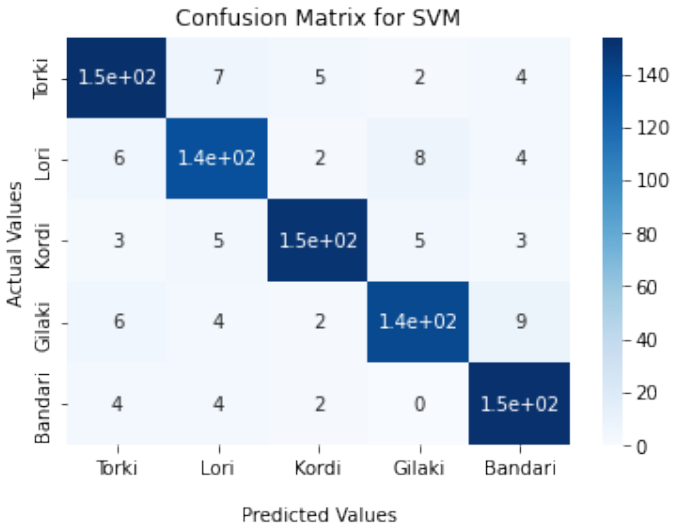
برای حل مشکل کمبود داده، بر روی داده‌ها دیتا آگمنتیشن انجام شد، به این صورت که هر آهنگ به ۵ بخش تقسیم شد و فیچرهای مورد نظر به صورت جداگانه از هر یک از این بخش‌ها استخراج شدند. با این کار عملا داده‌ها ۵ برابر شدند. همان طور که میدانیم، برخلاف تعداد فیچر، تعداد داده هر چقدر بیشتر باشد، کارایی مدل ما بهتر است. این تاثیر را می‌توان در مدل‌های پیاده سازی شده در ادامه دید.

۱.۵ classification :

نتایج حاصل از طبقه بندی با استفاده از داده‌های اضافه شده برای هر یک از الگوریتم‌ها در ادامه آورده شده است.

۱.۱.۵ SVM :

با در نظر گرفتن ترم Regularization = ۱۵ و کرنل rbf در الگوریتم soft-SVM نتایج زیر حاصل گشت.



شکل ۱۳: ماتریس کانفیوژن دست آمده برای الگوریتم SVM

accuracySVM: 0.8959608323133414

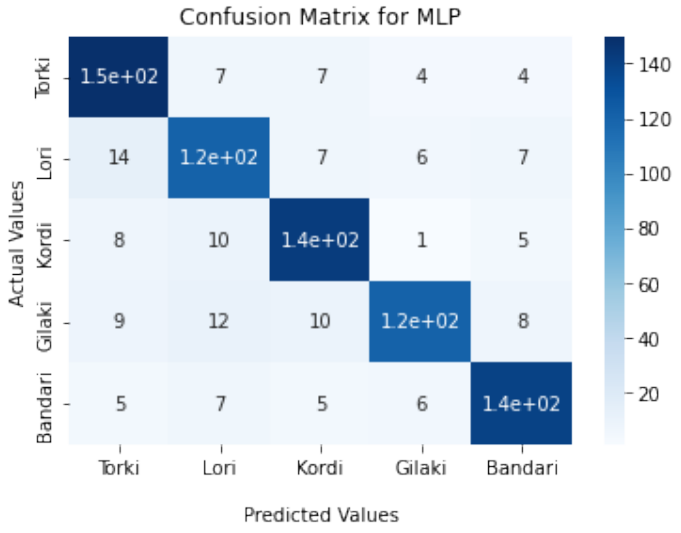
classificationSVMreport:

	precision	recall	f1-score	support
0.0	0.89	0.90	0.89	172
1.0	0.87	0.87	0.87	155
2.0	0.93	0.90	0.92	167
3.0	0.90	0.87	0.89	160
4.0	0.88	0.94	0.91	163
accuracy			0.90	817
macro avg	0.90	0.90	0.90	817
weighted avg	0.90	0.90	0.90	817

شکل ۱۴: نتایج حاصل به دست آمده برای الگوریتم SVM

۲.۱.۵ MLP :

ساختار شبکه در اینجا دو لایه با ۱۲۸ و ۳۲ نورون به همراه توابع فعالساز ReLu در نظر گرفته شد. نتایج این طبقه بند در ادامه آورده شده است.



شکل ۱۵: ماتریس کانفیوژن دست آمده برای الگوریتم MLP

accuracyMLP: 0.8261933904528764

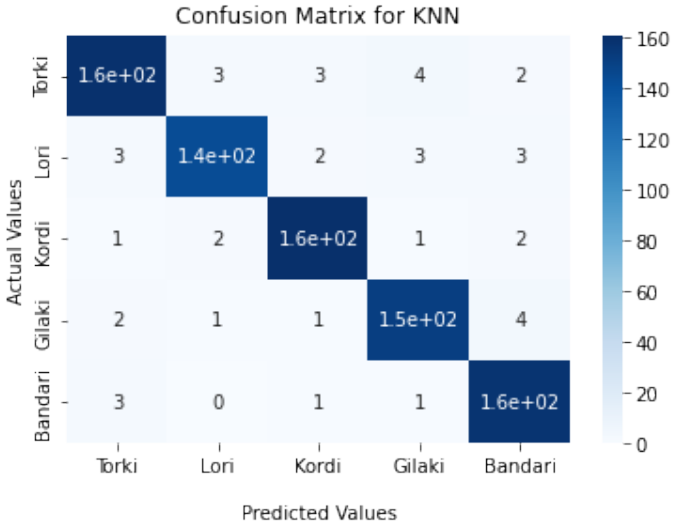
classificationMLPreport:

	precision	recall	f1-score	support
0.0	0.81	0.87	0.84	172
1.0	0.77	0.78	0.78	155
2.0	0.83	0.86	0.84	167
3.0	0.88	0.76	0.81	160
4.0	0.85	0.86	0.86	163
accuracy			0.83	817
macro avg	0.83	0.82	0.83	817
weighted avg	0.83	0.83	0.83	817

شکل ۱۶: نتایج حاصل به دست آمده برای الگوریتم MLP

۳.۱.۵ KNN :

نتایج الگوریتم KNN با در نظر گرفتن k=1 در ادامه آورده شده است.



شکل ۱۷: ماتریس کانفیوژن دست آمده برای الگوریتم KNN

accuracyKNN: 0.9485924112607099

classificationKNNreport:

	precision	recall	f1-score	support
0.0	0.95	0.93	0.94	172
1.0	0.96	0.93	0.94	155
2.0	0.96	0.96	0.96	167
3.0	0.94	0.95	0.95	160
4.0	0.93	0.97	0.95	163
accuracy			0.95	817
macro avg	0.95	0.95	0.95	817
weighted avg	0.95	0.95	0.95	817

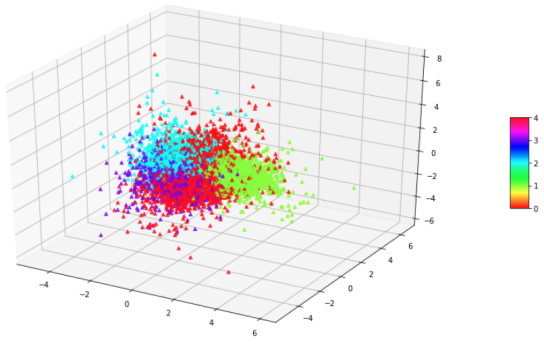
شکل ۱۸: نتایج حاصل به دست آمده برای الگوریتم KNN

میتوان دید که اضافه کردن داده تاثیر بسیار چشم گیری در عملکرد مدل‌ها دارد، به نحوی که به طور مثال در الگوریتم KNN توانستیم از دقت ۵۱ درصد به دقت ۹۵ درصد برسیم. این رشد چشمگیر در سایر الگوریتم‌های طبقه بندی نیز قابل رویت است.

۲.۵ clustering :

۱.۲.۵ K Means :

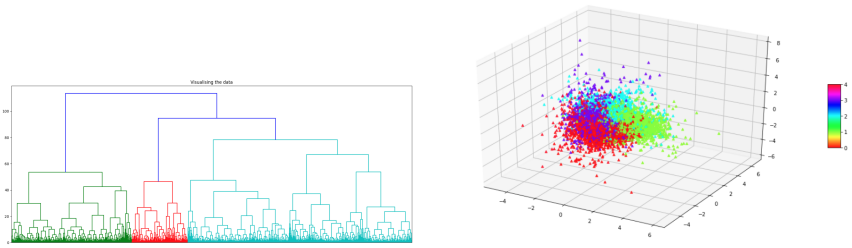
با همان پارامترهای تنظیم شده برای داده‌های ساده الگوریتم K Means را اجرا می‌کنیم نتایج مطابق شکل ۱۹ می‌باشند.



شکل ۱۹: نتیجه خوشه بندی با روش K Means

۲.۲.۵ : Hierarchical

در این قسمت نیز با معیارهای اندازه گیری خوشه ward، complete linkage و Average Linkage الگوریتم طبقه بندی bottom-up را اجرا می‌کنیم. و نتایج به شرح زیر می‌باشند.

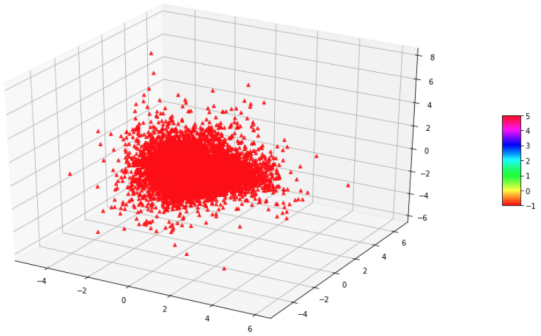


(آ) خوشه‌های به دست آمده (ب) Dendogram

شکل ۲۰: نتیجه خوشه بندی با معیار ward

۳.۲.۵ : DBSCAN

این الگوریتم نیز مجددا اجرا شده، اما به همان دلایلی که الگوریتم‌های خوشه بندی عملکرد خوبی در این پروژه نداشتند در این قسمت نیز دقتشان به حد قابل قبولی نرسید.

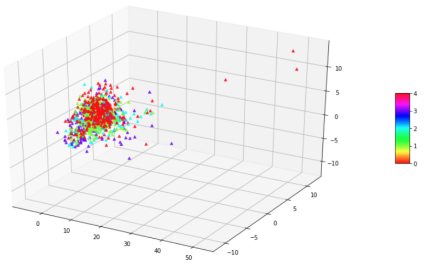


شکل ۲۱: نتیجه خوشه بندی با روش DBSCAN

Measure	K-Means	Hierarchical (Complete)	Hierarchical (Average)	Hierarchical (Ward)	DBSCAN
Silhouette	0.207	0.128	0.502	0.132	-0.44
Purity	0.228	0.131	0.104	0.167	0.177

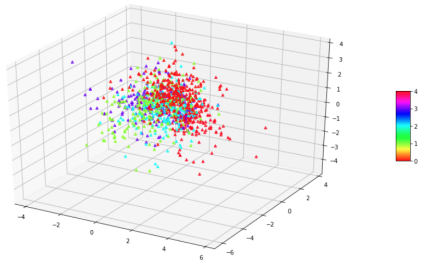
۶ نمایش داده ها در ابعاد پایین تر

برای اینکه بتوانیم کیفیت فیچرهای استخراج شده و کارایی آن‌ها برای انجام این پروژه داده‌ها را در فضای کاهش بعد یافته نمایش می‌دهیم تا به طور کیفی تفکیک پذیری آن‌ها با توجه به فیچرهای استخراج شده مشخص شود. برای این کار دو الگوریتم LDA و PCA را در طول ترم یاد گرفتیم. از آن جا که LDA با توجه به لیبل داده‌ها جهت‌های بهینه برای تصویر سازی را می‌یابد انتظار داریم داده‌ها پس از انتقال بر روی بردارهای به دست آمده از LDA تفکیک پذیری بیشتری نسبت به داده‌ها پس از انتقال با استفاده از PCA داشته باشند.
پس از کاهش بعد داده‌ها با استفاده از PCA بر روی سه کامپوننت اول نتایج زیر حاصل شد که به شکل سه بعدی قابل نشان دادن هستند.



شکل ۲۲: داده‌های آموزش پس از کاهش بعد با PCA

پس از کاهش بعد داده‌ها با استفاده از LDA بر روی سه بردار ویژه اول نتایج زیر حاصل شد که به شکل سه بعدی قابل نشان دادن هستند.



شکل ۲۳: داده‌های کاهش بعد یافته با استفاده از LDA